

VVV		VVV	MMM	MMM	SSSSSSSSSSSS	LLL	IIIIIIII	0000000000	
VVV		VVV	MMM	MMM	SSSSSSSSSSSS	LLL	IIIIIIII	0000000000	
VVV		VVV	MMM	MMM	SSSSSSSSSSSS	LLL	IIIIIIII	0000000000	
VVV		VVV	MMMMMM	MMMMMM	SSS	LLL	III	000	000
VVV		VVV	MMMMMM	MMMMMM	SSS	LLL	III	000	000
VVV		VVV	MMMMMM	MMMMMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSSSSSSSSS	LLL	III	0000000000	
VVV		VVV	MMM	MMM	SSSSSSSSSS	LLL	III	0000000000	
VVV		VVV	MMM	MMM	SSSSSSSSSS	LLL	III	0000000000	
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSSSSSSSSSSS	LLLLLLLLLLLLLLLL	IIIIIIII	0000000000	
VVV		VVV	MMM	MMM	SSSSSSSSSSSS	LLLLLLLLLLLLLLLL	IIIIIIII	0000000000	
VVV		VVV	MMM	MMM	SSSSSSSSSSSS	LLLLLLLLLLLLLLLL	IIIIIIII	0000000000	



```
CCCCCCCC  AAAAAA  LL      LL      IIIIII  MM      MM      AAAAAA  GGGGGGGG  EEEEEEEEE
CCCCCCCC  AAAAAA  LL      LL      IIIIII  MM      MM      AAAAAA  GGGGGGGG  EEEEEEEEE
CC        AA      AA  LL      LL      II      MMMM  MMMM  AA      AA  GG      GG      EE
CC        AA      AA  LL      LL      II      MMMM  MMMM  AA      AA  GG      GG      EE
CC        AA      AA  LL      LL      II      MM  MM  MM      AA      AA  GG      GG      EE
CC        AA      AA  LL      LL      II      MM      MM      AA      AA  GG      GG      EE
CC        AA      AA  LL      LL      II      MM      MM      AA      AA  GG      GG      EE
CC        AA      AA  LL      LL      II      MM      MM      AAAAAAAAAA  GG  GGGGGG  EE
CC        AAAAAAAAAA  LL      LL      II      MM      MM      AAAAAAAAAA  GG  GGGGGG  EE
CC        AAAAAAAAAA  LL      LL      II      MM      MM      AA      AA  GG      GG      EE
CC        AA      AA  LL      LL      II      MM      MM      AA      AA  GG      GG      EE
CC        AA      AA  LL      LL      II      MM      MM      AA      AA  GG      GG      EE
CCCCCCCC  AA      AA  LLLLLLLLLL  LLLLLLLLLL  IIIIII  MM      MM      AA      AA  GGGGGG  EEEEEEEEE
CCCCCCCC  AA      AA  LLLLLLLLLL  LLLLLLLLLL  IIIIII  MM      MM      AA      AA  GGGGGG  EEEEEEEEE

LL      IIIIII  SSSSSSSS
LL      IIIIII  SSSSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SSSSSS
LL      II      SSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LLLLLLLLLL  IIIIII  SSSSSSSS
LLLLLLLLLL  IIIIII  SSSSSSSS
```



CALLIMAGE  
Table of contents

Merge and call image

D 2

16-SEP-1984 02:16:39 VAX/VMS Macro V04-00

Page 0

(1)	3	copyright notice	
(1)	29	Program description	
(2)	66	Symbol Definitions	
(3)	75	Storage definitions	
(4)	86	lib\$adr_image	
(5)	112	lib\$map_image	
(6)	134	lib\$call_image	
(7)	251	search_mapped	Search mapped image list
(8)	284	add_mapped	Add to mapped image list
(9)	315	allocate	Allocate virtual memory
(10)	340	deallocate	Deallocate virtual memory



```

0000 1      .title  callimage      Merge and call image
0000 2      .ident  'V04-000'
0000 3      .sbttl  copyright notice
0000 4      :
0000 5      :*****
0000 6      :*
0000 7      :*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 8      :*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 9      :*  ALL RIGHTS RESERVED.
0000 10     :*
0000 11     :*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 12     :*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 13     :*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 14     :*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 15     :*  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 16     :*  TRANSFERRED.
0000 17     :*
0000 18     :*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 19     :*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 20     :*  CORPORATION.
0000 21     :*
0000 22     :*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 23     :*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 24     :*
0000 25     :*
0000 26     :*****
0000 27     :

```



```
0000 29      .sbtll Program description
0000 30      :
0000 31      Facility
0000 32      :
0000 33      LIB$CALL_IMAGE
0000 34      :
0000 35      Abstract
0000 36      :
0000 37      This procedure merges a specified image into the P0
0000 38      region and calls it at its transfer address. On return
0000 39      from the image, this procedure exits with the final
0000 40      status from the image. This procedure may be called
0000 41      more than once for the same image since the images will
0000 42      be retained in memory and simply re-entered on subsequent
0000 43      calls. The image must be PIC and re-entrant.
0000 44      :
0000 45      Environment
0000 46      :
0000 47      Native mode, User mode, Shared library routines
0000 48      :
0000 49      Author
0000 50      :
0000 51      Tim Halvorsen, February 1980
0000 52      :
0000 53      Modified by
0000 54      :
0000 55      V03-003 BLS0296      Benn Schreiber      7-APR-1984
0000 56      Correct call to lib$get_vm
0000 57      :
0000 58      V03-002 JWT0112      Jim Teague      18-Apr-1983
0000 59      Add LIB$ADR_IMAGE entry point.
0000 60      :
0000 61      V03-001 BLS0195      Benn Schreiber      19-Nov-1982
0000 62      Add LIB$MAP_IMAGE entry point.
0000 63      :
0000 64      :--
```



CALLIMAGE  
V04-000

Merge and call image  
Symbol Definitions

G 2

16-SEP-1984 02:16:39  
5-SEP-1984 04:38:45

VAX/VMS Macro V04-00  
[VMSLIB.SRC]CALLIMAGE.MAR;1

Page 3  
(2)

0000 66  
0000 67  
0000 68 :  
0000 69 :  
0000 70 :  
0000 71 :  
0000 72 :  
0000 73

.sbttl Symbol Definitions

Symbol Defintions

\$ihddef  
\$ihadef  
\$iacdef

: image header definitions  
: image activation transfer array  
: image activator flags



CALLIMAGE  
V04-000

Merge and call image  
Storage definitions

H 2

16-SEP-1984 02:16:39  
5-SEP-1984 04:38:45

VAX/VMS Macro V04-00  
[VMSLIB.SRC]CALLIMAGE.MAR;1

Page 4  
(3)

```
      0000 75      .sbttl Storage definitions
      0000 76
00000000 77      .psect _lib$data,noexe,wrt,pic
      0000 78      :
      0000 79      Writable storage definitions
      0000 80
      0000 81 list_head:
00000000 0000 82      .long 0 ; initialize list head to empty
      0004 83
00000000 84      .psect _lib$code,exe,nowrt,pic,shr
```



```
0000 86 .sbtll lib$adr_image
0000 87 :---
0000 88 :
0000 89 :   This procedure maps an image into P0 space and returns the
0000 90 :   base address of the image to the caller. R6 is used as a
0000 91 :   logical flag -- if 1, then return base address of image.
0000 92 :   Note that this entry point will always set r6 -- other
0000 93 :   entry points will clear it.
0000 94 :
0000 95 :   Inputs:
0000 96 :       4(ap) = Address of descriptor of image file specification
0000 97 :       8(ap) = Address of descriptor of default file specification
0000 98 :       12(ap) = Address of longword to return image base address
0000 99 :
0000 100 :   Outputs:
0000 101 :
0000 102 :       R0 = Status returned by $IMGACT
0000 103 :---
OFFC 0000 104 .entry lib$adr_image -
0002 105      ^m<r2,r3,r4,r5,r6,r7,r8,r9,r10,r11>
0002 106      ifnowrt #4,a12(ap),accvio      ; puke if can't write base address
56 5B D4 0009 107      clr r11      ; just activate - don't call
01 D0 000B 108      movl #1,r6      ; want base address, not transfer address
16 11 000E 109      brb do_image      ; branch to common processing
0010 110
```



```
0010 112 .sbtll lib$map_image
0010 113 ---
0010 114 This procedure maps an image into P0 space and returns the
0010 115 transfer address of the image to the caller.
0010 116
0010 117 Inputs:
0010 118
0010 119 4(ap) = Address of descriptor of image file specification
0010 120 8(ap) = Address of descriptor of default file specification
0010 121 12(ap) = Address of longword to return transfer address
0010 122
0010 123 Outputs:
0010 124
0010 125 R0 = Status returned by $IMGACT
0010 126 ---
0010 127 .entry lib$map_image,-
0012 128 ^m<r2,r3,r4,r5,r6,r7,r8,r9,r10,r11>
5B D4 0012 129 clrl r11 ;flag to image activate only
0014 130 ifnowrt #4,@12(ap),accvio ;branch if unable to write result
56 D4 001B 131 clrl r6 ;not interested in base address
07 11 001D 132 brb do_image ;branch to common processing
```



```
001F 134 .sbtll lib$call_image
001F 135
001F 136
001F 137 This procedure merges a specified image into the P0
001F 138 region and calls it at its transfer address. On return
001F 139 from the image, this procedure exits with the final
001F 140 status from the image. This procedure may be called
001F 141 more than once for the same image since the images will
001F 142 be retained in memory and simply re-entered on subsequent
001F 143 calls. The image must be PIC and re-entrant.
001F 144
001F 145 Inputs:
001F 146
001F 147 04(ap) = Address of descriptor of image file specification
001F 148 08(ap) = Address of descriptor of default file specification
001F 149 12(ap) = Address of argument list to image (passed to image
001F 150 as the 7th argument -- 28(ap))
001F 151
001F 152 Outputs:
001F 153
001F 154 r0 = Status returned by image or by $IMGACT
001F 155
001F 156
001F 157
001F 158 .enabl lsb
001F 159
001F 159 .entry lib$call_image,-
0021 160 ^m<r2,r3,r4,r5,r6,r7,r8,r9,r10,r11>
0021 161
0021 162 movl #1,r11 ; flag to call image also
0024 163 clrl r6 ; don't want image base address
0026 164 do_image:
0026 165 movab list_head,r10 ; r10 = Address of section list head
002D 166 ifnord #8,@4(ap),accvio ; branch if unable to read descriptor
0034 167 movq @4(ap),r2 ; get descriptor of image name
0038 168 movzwl r2,r2 ; and isolate length from it
003B 169 ifrd r2,(r3),10$ ; branch if have read access
0041 170 accvio: movzwl #ss$_accvio,r0 ; cannot access descriptor
0046 171 ret
0047 172
0047 173 10$: bsbw search_mapped ; search mapped image list
004A 174 blbs r0,20$ ; if found, don't map it
004D 175
004D 176 this is the first time we have invoked this image.
004D 177 merge the image into existing p0 address space.
004D 178
004D 179
004D 179 ifnord #8,@8(ap),accvio ; branch if unable to read descriptor
0054 180 movq @8(ap),-(sp) ; put default descriptor on stack
0058 181 movq r2,-(sp) ; put input descriptor on stack
005B 182 movq #1,-(sp) ; specify p0 region
005E 183 movl #512,r1 ; length of image activator work area
0065 184 bsbw allocate ; allocate a page for it
0068 185 movl r0,r7 ; save address of work area
006B 186 movl sp,r1 ; get address of stack
006E 187 $imgact_s name=8(r1),- ; merge image into address space
006E 188 dflnam=16(r1),- ; using supplied default as default
006E 189 hdrbuf=(r7),- ; address of work area
006E 190 imgctl=#iac$m_merge!iac$m_expreg,- ; merge into region
```

5A 00000000'EF 9E 0026 165  
52 04 BC 7D 0034 167  
52 52 3C 0038 168  
50 0000'8F 3C 0041 170  
00B3 30 0047 173  
68 50 E8 004A 174  
7E 08 BC 7D 0054 180  
7E 52 7D 0058 181  
7E 01 7D 005B 182  
51 00000200 8F D0 005E 183  
00E0 30 0065 184  
57 50 D0 0068 185  
51 5E D0 006B 186  
006E 187  
006E 188  
006E 189  
006E 190



```
006E 191      inadr=(r1),-      ; address range to map into
006E 192      retadr=(r1)      ; address to return range to
5B 50 E9 0087 193      blbc     r0, maperr      ; branch if error
008A 194      $imgfix_s      ; perform address relocation
51 50 E9 0091 195      blbc     r0, maperr      ; quit if error occurred
03 56 E9 0094 196      blbc     r6, 15$      ; don't need image base address
55 6E D0 0097 197      movl     (sp), r5      ; do need base address - save it
58 67 7D 009A 198 15$:  movq     (r7), r8      ; r8 = address of ihd (header desc)
009D 199      movzwl    ihd$w_activoff(r8), r0 ; r9 = address of ifd (file desc)
50 02 A8 3C 009D 200      movzwl    ihd$w_activoff(r8), r0 ; get offset to activation data
51 68 40 9E 00A1 201      movab     iha$L_tfradr1(r8)[r0], r1 ; address of transfer address array
7C 10 00A5 202      bsbb      add_mapped      ; add to mapped image list
5E 18 C0 00A7 203      addl     #24, sp      ; deallocate descriptors
08 56 E9 00AA 204      blbc     r6, 20$      ; don't want base address - continue
0C BC 55 D0 00AD 205      movl     r5, @12(ap) ; write saved base address
50 01 D0 00B1 206      movl     #1, r0      ; success
04 00B4 207      ret
00B5 208      ;
00B5 209      ; call the image
00B5 210      ;
58 67 7D 00B5 211 20$:  movq     (r7), r8      ; address of ihd (image header desc)
50 02 A8 3C 00B8 212      movzwl    ihd$w_activoff(r8), r0 ; get offset to activation data
51 68 40 9E 00BC 213      movab     iha$L_tfradr1(r8)[r0], r1 ; address of transfer address array
1A 5B E9 00C0 214      blbc     r11, 60$      ; branch if map image call
5D DD 00C3 215      pushl     fp      ; save FP from destruction
0C AC DD 00C5 216      pushl     12(ap)      ; address of user argument list
00 DD 00C8 217      pushl     #0      ; no cli option flags
20 A8 DD 00CA 218      pushl     ihd$L_lnkflags(r8) ; pass thru link option flags
7E 58 7D 00CD 219      movq     r8, -(sp)      ; address of ihd and ifd
F8 AF 9F 00D0 220      pushab    b^cllicallback ; address of dummy call back routine
51 DD 00D3 221      pushl     r1      ; address of transfer address array
00 B1 07 FB 00D5 222      calls     #7, a(r1) ; call native mode code
5D 8ED0 00D9 223      popl     fp      ; restore saved frame pointer
04 00DC 224      ret
00DD 225      ;
00DD 226      ; map image only. return transfer address to caller
00DD 227      ;
0C BC 61 D0 00DD 228 60$:  movl     (r1), @12(ap) ; return address to caller
50 01 D0 00E1 229      movl     #1, r0      ; set status
04 00E4 230      ret
00E5 231      ;
00E5 232      ; error from $imgact call - map to ie.nsf (no such file)
00E5 233      ;
50 50 DD 00E5 234 maperr: pushl     r0      ; save status over deallocate
50 57 D0 00E7 235      movl     r7, r0      ; address of work page
00000200 8F D0 00EA 236      movl     #512, r1      ; length of work page
006A 30 00F1 237      bsbw     deallocate      ; deallocate it
50 8ED0 00F4 238      popl     r0      ; restore status
04 00F7 239      ret
00F8 240      ;
00F8 241      ; dummy cli call back routine
00F8 242      ;
00F8 243      ;
00F8 244      clicallback:
50 0000 00F8 245      .word     0
04 D4 00FA 246      clrl     r0      ; return failure
04 00FC 247      ret
```



CALLIMAGE  
V04-000

Merge and call image  
lib\$call\_image

M 2

16-SEP-1984 02:16:39  
5-SEP-1984 04:38:45

VAX/VMS Macro V04-00  
[VMSLIB.SRC]CALLIMAGE.MAR;1

Page 9  
(6)

00FD 248  
00FD 249

.dsabl lsb



```
00FD 251 .sbttl search_mapped Search mapped image list
00FD 252 :---
00FD 253 :
00FD 254 This routine searches the mapped image list to
00FD 255 determine if the image has already been mapped
00FD 256 :
00FD 257 Inputs:
00FD 258
00FD 259 r2/r3 = Descriptor of image file specification
00FD 260 r10 = Address of mapped image list head
00FD 261 :
00FD 262 Outputs:
00FD 263
00FD 264 r0 = true if already in memory
00FD 265 if found, r7 = address of imgact work page
00FD 266 :
00FD 267 :---
00FD 268 search_mapped:
57 6A D0 00FD 269 movl (r10),r7 ; address of first entry
50 08 A7 9A 0100 270 10$: beql 50$ ; branch if not found
09 A7 50 2D 0102 271 movzbl 8(r7),r0 ; length of string name
0C BB 0106 272 pushr #^m<r2,r3> ; save registers
63 52 00 09 A7 50 2D 0108 273 cmpc5 r0,9(r7),#0,r2,(r3) ; string match?
0C BA 010F 274 popr #^m<r2,r3> ; restore registers
57 05 13 0111 275 beql 20$ ; branch if found
57 67 D0 0113 276 movl (r7),r7 ; next entry in list
57 04 A7 D0 0116 277 brb 10$ ; and continue searching
50 01 D0 0118 278 20$: movl 4(r7),r7 ; return address of imgact work page
05 05 011C 279 movl #1,r0 ; success
50 D4 0120 280 rsb ;
05 0122 281 50$: clrl r0 ; failure
05 0122 282 rsb
```



```
0123 284 .sbtll add_mapped Add to mapped image list
0123 285 :---
0123 286 :
0123 287 Add the image just mapped to the mapped image list
0123 288 :
0123 289 Inputs:
0123 290 :
0123 291 r2/r3 = Descriptor of image file specification
0123 292 r7 = Address of imgact work page
0123 293 r10 = Address of mapped image list head
0123 294 :
0123 295 Outputs:
0123 296 :
0123 297 None
0123 298 :
0123 299 :---
0123 300 add_mapped:
0123 301 pushr #^m<r1,r2,r3> ; save address, descriptor
51 52 3C 0125 302 movzwl r2,r1 ; extract image name length
51 09 C0 0128 303 addl #9,r1 ; compute list entry length
001A 30 012B 304 bsbw allocate ; allocate memory
0E BA 012E 305 popr #^m<r1,r2,r3> ; restore address, descriptor
60 6A D0 0130 306 movl (r10),(r0) ; insert into list
6A 50 D0 0133 307 movl r0,(r10)
04 A0 57 D0 0136 308 movl r7,4(r0) ; store address of imgact work page
08 A0 52 90 013A 309 movb r2,8(r0) ; store length of counted string
09 A0 63 3E BB 013E 310 pushr #^m<r1,r2,r3,r4,r5> ; save registers
52 28 0140 311 movc r2,(r3),9(r0) ; store string itself
3E BA 0145 312 popr #^m<r1,r2,r3,r4,r5> ; restore registers
05 0147 313 rsb
```



```
0148 315 .sbtll allocate Allocate virtual memory
0148 316 :---
0148 317 :
0148 318 Allocate virtual memory
0148 319 :
0148 320 Inputs:
0148 321 :
0148 322 r1 = Number of bytes to allocate
0148 323 :
0148 324 Outputs:
0148 325 :
0148 326 r0 = Address of allocated storage
0148 327 :
0148 328 If error obtaining memory, a RET is performed with the status
0148 329 :---
0148 330 allocate:
0148 331 pushl r1 ; Get length onto stack
0148 332 pushal -(sp) ; Address of lw to put address
0148 333 pushab 8(sp) ; Address of lw containing length
0148 334 calls #2,g^lib$get_vm ; Obtain virtual memory
0148 335 blbc r0,90$ ; branch if error
0148 336 movq (sp)+,r0 ; Return address of memory
0148 337 rsb
0148 338 90$: ret
```

51 DD 0148  
7E DF 014A  
08 AE 9F 014C  
00000000'GF 02 FB 014F  
04 50 E9 0156  
50 8E 7D 0159  
05 015C  
04 015D



```
015E 340 .sbttl deallocate Deallocate virtual memory
015E 341 :---
015E 342 :
015E 343 Deallocate virtual memory
015E 344 :
015E 345 Inputs:
015E 346 :
015E 347 r0 = Address of memory to deallocate
015E 348 r1 = Number of bytes to deallocate
015E 349 :
015E 350 Outputs:
015E 351 :
015E 352 If error detected, a RET is performed with the status
015E 353 :
015E 354 :---
015E 355 deallocate:
015E 356 movq r0,-(sp) ; put parameters on stack
015E 357 pushl sp ; address of lw containing address
015E 358 pushab 8(sp) ; address of lw containing length
015E 359 calls #2,g^lib$free_vm ; deallocate virtual memory
015E 360 addl2 #8,sp ; clear stack
015E 361 blbc r0,90$ ; branch if error
015E 362 rsb ;
015E 363 90$: ret ; return with status
```

7E 50 7D 015E 356  
5E DD 0161 357  
08 AE 9F 0163 358  
00000000 GF 02 FB 0166 359  
5E 08 C0 016D 360  
01 50 E9 0170 361  
05 0173 362  
04 0174 363 90\$:



CALLIMAGE  
V04-000

E 3  
Merge and call image  
deallocate Deallocate virtual memory

16-SEP-1984 02:16:39 VAX/VMS Macro V04-00 Page 14  
5-SEP-1984 04:38:45 [VMSLIB.SRC]CALLIMAGE.MAR;1 (12)

0175 365  
0175 366 .end



CALLIMAGE  
Symbol table

Merge and call image

F 3

16-SEP-1984 02:16:39  
5-SEP-1984 04:38:45

VAX/VMS Macro V04-00  
[VMSLIB.SRC]CALLIMAGE.MAR;1

Page 15  
(12)

\$ST1	= 00000000		
ACCVIO	00000041	R	03
ADD_MAPPED	00000123	R	03
ALLOCATE	00000148	R	03
CLICALLBACK	000000F8	R	03
DEALLOCATE	0000015E	R	03
DO_IMAGE	00000026	R	03
IACSM_EXPREG	= 00000020		
IACSM_MERGE	= 00000010		
IHASL_TFRADR1	= 00000000		
IHDSL_LNKFLAGS	= 00000020		
IHDSW_ACTIVOFF	= 00000002		
LIB\$ADR_IMAGE	00000000	RG	03
LIB\$CALC_IMAGE	0000001F	RG	03
LIB\$FREE_VM	*****	X	03
LIB\$GET_VM	*****	X	03
LIB\$MAP_IMAGE	00000010	RG	03
LIST HEAD	00000000	R	02
MAPERR	000000E5	R	03
SEARCH_MAPPED	000000FD	R	03
SS\$ ACCVIO	*****	X	03
SY\$IMGACT	*****	GX	03
SY\$IMGFIX	*****	GX	03

+-----+  
! Psect synopsis !  
+-----+

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 ( 0.)	00 ( 0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
\$ABSS	00000000 ( 0.)	01 ( 1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
_LIB\$DATA	00000004 ( 4.)	02 ( 2.)	PIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC BYTE
_LIB\$CODE	00000175 ( 373.)	03 ( 3.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC BYTE

+-----+  
! Performance indicators !  
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	32	00:00:00.09	00:00:00.67
Command processing	123	00:00:00.58	00:00:02.78
Pass 1	156	00:00:02.54	00:00:05.72
Symbol table sort	0	00:00:00.14	00:00:00.33
Pass 2	77	00:00:00.85	00:00:01.90
Symbol table output	4	00:00:00.03	00:00:00.23
Psect synopsis output	3	00:00:00.03	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	397	00:00:04.27	00:00:11.68

The working set limit was 1050 pages.  
12793 bytes (25 pages) of virtual memory were used to buffer the intermediate code.  
There were 10 pages of symbol table space allocated to hold 129 non-local and 9 local symbols.  
366 source lines were read in Pass 1, producing 21 object records in Pass 2.  
17 pages of virtual memory were used to define 16 macros.



+-----+  
! Macro library statistics !  
+-----+

Macro library name

Macros defined

-----  
\$255\$DUA28:[SYS.OBJ]LIB.MLB;1  
\$255\$DUA28:[SYSLIB]STARLET.MLB;2  
TOTALS (all libraries)

-----  
5  
8  
13

227 GETS were required to define 13 macros.

There were no errors, warnings or information messages.

MACRO/DISA=TRACE/LIS=LISS:CALLIMAGE/OBJ=OBJ\$:CALLIMAGE MSRC\$:CALLIMAGE/UPDATE=(ENH\$:CALLIMAGE)+EXECMLS/LIB



0435 AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY

